MAGAZIN ONLINE

proiect realizat de:Poovnicu David Stefan

grupa:242

Facultatea de Matematica si Informatica

Universitatea din Bucuresti

2024-2025

1. Descrierea modelului real, a utilității și a regulilor de funcționare

Baza de date pentru un magazin online urmărește gestionarea eficientă a produselor disponibile spre vânzare, a clienților, a comenzilor plasate, a stocurilor și a plăților. Fiecare client poate realiza comenzi care pot conține unul sau mai multe produse. Produsele sunt organizate pe categorii. Fiecare comandă este asociată cu o plată, iar stocul fiecărui produs este monitorizat pentru a evita vânzarea produselor indisponibile.

Reguli de funcționare:

- Un client trebuie să fie înregistrat pentru a realiza o comandă.
- Un produs trebuie să fie în stoc pentru a putea fi comandat.
- O comandă poate conține mai multe produse, în diverse cantități.
- Plata unei comenzi se face integral, la finalizarea comenzii.
- Se păstrează istoricul comenzilor și al plăților.

2. Prezentarea constrângerilor (restricții, reguli impuse asupra modelului)

- Un produs nu poate fi vândut dacă nu există în stoc Cantitate_Disponibila >= numărul comandat.
- Un client poate plasa oricând mai multe comenzi, fără limită de număr.
- O comandă nu poate conține produse duplicate (același produs de două ori în aceeași comandă).
- Fiecare produs aparține unei singure categorii.
- Fiecare comandă trebuie să fie asociată cu o plată.
- Email-ul clientului trebuie să fie unic.
- Pentru fiecare plată, suma să corespundă totalului comenzii asociate.

3. Descrierea entităților (tabele) și precizarea cheii primare

1. **Client**
    a. Cheie primară: ID_Client
    b. Descriere: Reprezintă utilizatorii magazinului online.
2. **Produs**
    a. Cheie primară: ID_Produs
    b. Descriere: Reprezintă produsele disponibile pentru vânzare.
3. **Categorie**
    a. Cheie primară: ID_Categorie
    b. Descriere: Reprezintă categoriile de produse.
4. **Comanda**
    a. Cheie primară: ID_Comanda
    b. Descriere: Reprezintă comenzile plasate de clienți.
5. **Detalii_Comanda** (tabel asociativ)
    a. Cheie primară compusă: (ID_Comanda, ID_Produs)
    b. Descriere: Reprezintă legătura N:M dintre comenzi și produse, cu detalii despre cantitate și preț.
6. **Stoc**
    a. Cheie primară: ID_Produs
    b. Descriere: Reprezintă stocul disponibil pentru fiecare produs.
7. **Plata**
    a. Cheie primară: ID_Plata
    b. Descriere: Reprezintă plățile efectuate pentru comenzi.

4. Descrierea relațiilor, inclusiv precizarea cardinalității acestora

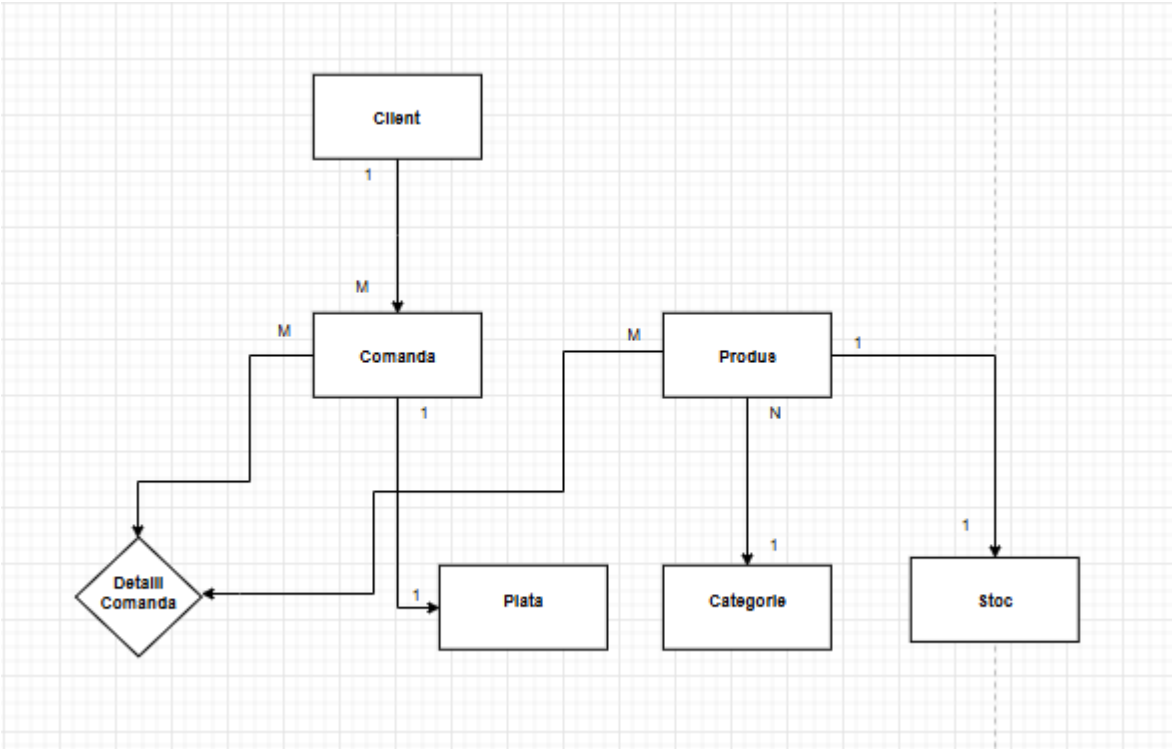
- **Client – Comanda:** 1:N
  Un client poate plasa mai multe comenzi, dar o comandă aparține unui singur client.
- **Comanda – Detalii_Comanda – Produs:** N:M
  O comandă poate conține mai multe produse, iar un produs poate fi inclus în mai multe comenzi. Relația este implementată prin tabelul asociativ Detalii_Comanda.
- **Produs – Categorie:** N:1
  Fiecare produs aparține unei singure categorii, dar o categorie poate conține mai multe produse.
- **Produs – Stoc:** 1:1
  Fiecare produs are un singur stoc asociat.
- **Comanda – Plata:** 1:1
  Fiecare comandă are o singură plată asociată.

5. Descrierea atributelor, tipul de date și eventuale constrângeri

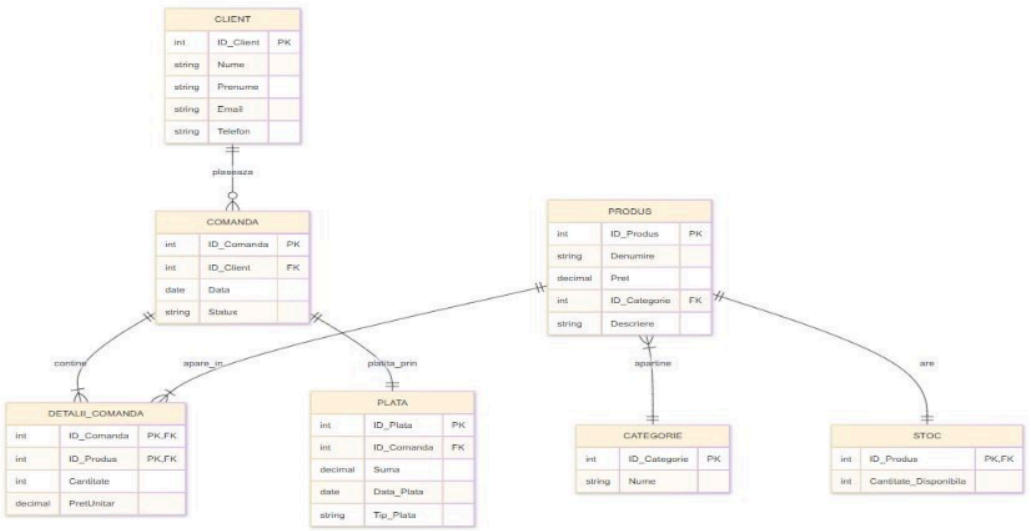
- **Client:**
  - ID_Client (int, PK, auto-increment)
  - Nume (varchar, not null)
  - Prenume (varchar, not null)
  - Email (varchar, not null, unic)
  - Telefon (varchar, optional)
- **Produs:**
  - ID_Produs (int, PK, auto-increment)
  - Denumire (varchar, not null)
  - Pret (decimal, not null)
  - ID_Categorie (int, FK, not null)
  - Descriere (varchar, optional)
- **Categorie:**
  - ID_Categorie (int, PK, auto-increment)
  - Nume (varchar, not null)
- **Comanda:**
  - ID_Comanda (int, PK, auto-increment)
  - ID_Client (int, FK, not null)
  - Data (date, not null)
  - Status (varchar, not null, ex: "In procesare", "Finalizata")

- **Detalii_Comanda:**
  - ID_Comanda (int, PK, FK, not null)
  - ID_Produs (int, PK, FK, not null)
  - Cantitate (int, not null)
  - PretUnitar (decimal, not null)
- **Stoc:**
  - ID_Produs (int, PK, FK, not null)
  - Cantitate_Disponibila (int, not null, >=0)
- **Plata:**
  - ID_Plata (int, PK, auto-increment)
  - ID_Comanda (int, FK, not null, unic)
  - Suma (decimal, not null)
  - Data_Plata (date, not null)
  - Tip_Plata (varchar, not null, ex: "Card", "Transfer", "Ramburs")

6. Realizarea diagramei entitate-relație corespunzătoare descrierii de la punctele 3-5.



7. Realizarea diagramei conceptuale corespunzătoare diagramei entitate-relație proiectate la punctul 6. Diagrama conceptuală obținută trebuie să conțină minimum 7 tabele (fără considerarea subentităților), dintre care cel puțin un tabel asociativ



8. Enumerarea schemelor relaționale corespunzătoare diagramei conceptuale proiectate la punctul 7.

CLIENT(#id_client, nume, email, parola, data_inregistrare)

PRODUS(#id_produs, nume, pret, id_categorie, descriere)

CATEGORIE(#id_categorie, nume_categorie)

COMANDA(#id_comanda, data, id_client, adresa_livrare, status)

DETALII_COMANDA(#id_comanda, #id_produs, cantitate, pret_unitar)

STOC(#id_produs, cantitate_disponibila)

PLATA(#id_plata, id_comanda, suma, data_plata, tip_plata)

9. Realizarea normalizării până la forma normală 3 (FN1-FN3).

Exemplu non-FN1 → FN1

non-FN1:

COMANDA(id_comanda, produse_comandate, data, id_client)

unde *produse_comandate* = listă de produse (de ex: "produs1, produs2, produs3")

Transformare în FN1:

Dezvoltăm atributele cu valori atomice, deci facem o tabelă separată pentru detaliile comenzii:

COMANDA(id_comanda, data, id_client)

DETALII_COMANDA(id_comanda, id_produs, cantitate, pret_unitar)

Exemplu non-FN2 → FN2

non-FN2:

DETALII_COMANDA(id_comanda, id_produs, nume_produs, cantitate, pret_unitar)

Cheia primară: (id_comanda, id_produs)

nume_produs depinde doar de id_produs, nu de cheia întreagă.

Transformare în FN2:

Separăm atributele care depind doar de o parte a cheii:

DETALII_COMANDA(id_comanda, id_produs, cantitate, pret_unitar)

PRODUS(id_produs, nume_produs, ...)

Exemplu non-FN3 → FN3

non-FN3:

PRODUS(id_produs, nume_produs, id_categorie, nume_categorie)

nume_categorie depinde de id_categorie, nu direct de cheia primară id_produs (dep. tranzitivă).

Transformare în FN3:

Separăm categoria într-o relație distinctă:

PRODUS(id_produs, nume_produs, id_categorie)

CATEGORIE(id_categorie, nume_categorie)

10. Crearea unei secvențe ce va fi utilizată în inserarea înregistrărilor în tabele (punctul 11).

```
Sequence SEQ_ID created.
```

```
1   CREATE SEQUENCE seq_id START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;
2
```

11. Crearea tabelelor în SQL și inserarea de date coerente în fiecare dintre acestea (minimum 5 înregistrăr Fie care tabel neasociativ; minimum 10 înregistrări în tabelele asociative; maxim 30 de înregistrări în tabel.

10.SQL > ...

```sql
CREATE TABLE CLIENT (
    id_client NUMBER PRIMARY KEY,
    nume VARCHAR2(50),
    email VARCHAR2(50),
    parola VARCHAR2(50),
    data_inregistrare DATE
);

CREATE TABLE CATEGORIE (
    id_categorie NUMBER PRIMARY KEY,
    nume_categorie VARCHAR2(50)
);

CREATE TABLE PRODUS (
    id_produs NUMBER PRIMARY KEY,
    nume VARCHAR2(50),
    pret NUMBER(10,2),
    id_categorie NUMBER,
    descriere VARCHAR2(100),
    CONSTRAINT fk_produs_categorie FOREIGN KEY (id_categorie) REFERENCES CATEGORIE(id_categorie)
);

CREATE TABLE COMANDA (
    id_comanda NUMBER PRIMARY KEY,
    data DATE,
    id_client NUMBER,
    adresa_livrare VARCHAR2(100),
    status VARCHAR2(20),
    CONSTRAINT fk_comanda_client FOREIGN KEY (id_client) REFERENCES CLIENT(id_client)
);

CREATE TABLE STOC (
    id_produs NUMBER PRIMARY KEY,
    cantitate_disponibila NUMBER,
    CONSTRAINT fk_stoc_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs)
);

CREATE TABLE PLATA (
    id_plata NUMBER PRIMARY KEY,
    id_comanda NUMBER,
    valoare NUMBER(10,2),
    data_plata DATE,
    tip_plata VARCHAR2(20),
    CONSTRAINT fk_plata_comanda FOREIGN KEY (id_comanda) REFERENCES COMANDA(id_comanda)
);

CREATE TABLE DETALII_COMANDA (
    id_comanda NUMBER,
    id_produs NUMBER,
    cantitate NUMBER,
    pret_unitar NUMBER(10,2),
    PRIMARY KEY (id_comanda, id_produs),
    CONSTRAINT fk_detalii_comanda FOREIGN KEY (id_comanda) REFERENCES COMANDA(id_comanda),
    CONSTRAINT fk_detalii_produs FOREIGN KEY (id_produs) REFERENCES PRODUS(id_produs)
);
```

```
Table CLIENT created.


Table CATEGORIE created.


Table PRODUS created.


Table COMANDA created.


Table STOC created.


Table PLATA created.
```

```sql
-- CATEGORIE
INSERT INTO CATEGORIE VALUES (1, 'Electronice');
INSERT INTO CATEGORIE VALUES (2, 'Haine');
INSERT INTO CATEGORIE VALUES (3, 'Carti');
INSERT INTO CATEGORIE VALUES (4, 'Jucarii');
INSERT INTO CATEGORIE VALUES (5, 'Electrocasnice');

-- CLIENT
INSERT INTO CLIENT VALUES (1, 'Popescu Ana', 'ana.popescu@email.com', 'parola1', TO_DATE('2024-01-10', 'YYYY-MM-DD
INSERT INTO CLIENT VALUES (2, 'Ionescu Mihai', 'mihai.ionescu@email.com', 'parola2', TO_DATE('2024-02-15', 'YYYY-M
INSERT INTO CLIENT VALUES (3, 'Stan Maria', 'maria.stan@email.com', 'parola3', TO_DATE('2024-03-20', 'YYYY-MM-DD')
INSERT INTO CLIENT VALUES (4, 'Dumitru Vlad', 'vlad.dumitru@email.com', 'parola4', TO_DATE('2024-04-25', 'YYYY-MM-
INSERT INTO CLIENT VALUES (5, 'Georgescu Raluca', 'raluca.georgescu@email.com', 'parola5', TO_DATE('2024-05-30', '

-- PRODUS
INSERT INTO PRODUS VALUES (1, 'Laptop Lenovo', 3500.00, 1, 'Laptop performant pentru birou');
INSERT INTO PRODUS VALUES (2, 'Rochie vara', 120.50, 2, 'Rochie din bumbac, marimea M');
INSERT INTO PRODUS VALUES (3, 'Harry Potter', 60.00, 3, 'Carte fantasy populara');
INSERT INTO PRODUS VALUES (4, 'Masinuta', 35.00, 4, 'Jucarie pentru copii peste 3 ani');
INSERT INTO PRODUS VALUES (5, 'Mixer Philips', 240.00, 5, 'Mixer de bucatarie cu 5 viteze');

-- COMANDA
INSERT INTO COMANDA VALUES (1, TO_DATE('2024-06-01', 'YYYY-MM-DD'), 1, 'Str. Lalelelor 5, Bucuresti', 'livrata');
INSERT INTO COMANDA VALUES (2, TO_DATE('2024-06-03', 'YYYY-MM-DD'), 2, 'Bd. Libertatii 10, Timisoara', 'procesare'
INSERT INTO COMANDA VALUES (3, TO_DATE('2024-06-05', 'YYYY-MM-DD'), 3, 'Str. Florilor 8, Cluj', 'anulata');
INSERT INTO COMANDA VALUES (4, TO_DATE('2024-06-07', 'YYYY-MM-DD'), 4, 'Aleea Teiului 18, Iasi', 'livrata');
INSERT INTO COMANDA VALUES (5, TO_DATE('2024-06-09', 'YYYY-MM-DD'), 5, 'Str. Zorilor 2, Constanta', 'procesare');

-- STOC
INSERT INTO STOC VALUES (1, 10);
INSERT INTO STOC VALUES (2, 22);
INSERT INTO STOC VALUES (3, 15);
INSERT INTO STOC VALUES (4, 30);
INSERT INTO STOC VALUES (5, 5);

-- PLATA
INSERT INTO PLATA VALUES (1, 1, 3500.00, TO_DATE('2024-06-01', 'YYYY-MM-DD'), 'card');
INSERT INTO PLATA VALUES (2, 2, 120.50, TO_DATE('2024-06-03', 'YYYY-MM-DD'), 'ramburs');
INSERT INTO PLATA VALUES (3, 3, 60.00, TO_DATE('2024-06-05', 'YYYY-MM-DD'), 'card');
INSERT INTO PLATA VALUES (4, 4, 35.00, TO_DATE('2024-06-07', 'YYYY-MM-DD'), 'card');
INSERT INTO PLATA VALUES (5, 5, 240.00, TO_DATE('2024-06-09', 'YYYY-MM-DD'), 'ramburs');

-- DETALII_COMANDA
INSERT INTO DETALII_COMANDA VALUES (1, 1, 1, 3500.00);
INSERT INTO DETALII_COMANDA VALUES (2, 2, 2, 120.50);
INSERT INTO DETALII_COMANDA VALUES (2, 3, 1, 60.00);
INSERT INTO DETALII_COMANDA VALUES (3, 3, 1, 60.00);
INSERT INTO DETALII_COMANDA VALUES (3, 4, 2, 35.00);
INSERT INTO DETALII_COMANDA VALUES (4, 4, 2, 35.00);
INSERT INTO DETALII_COMANDA VALUES (4, 5, 1, 240.00);
INSERT INTO DETALII_COMANDA VALUES (5, 1, 1, 3500.00);
INSERT INTO DETALII_COMANDA VALUES (5, 2, 1, 120.50);
INSERT INTO DETALII_COMANDA VALUES (5, 5, 1, 240.00);
```

```
1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.

1 row inserted.
```

```
ID_CLIENT NUME                  EMAIL                        PAROLA    DATA_INR
--------- --------------------- ---------------------------- --------- --------
        1 Popescu Ana           ana.popescu@email.com        parola1   10/01/24
        2 Ionescu Mihai         mihai.ionescu@email.com      parola2   15/02/24
        3 Stan Maria            maria.stan@email.com         parola3   20/03/24
        4 Dumitru Vlad          vlad.dumitru@email.com       parola4   25/04/24
        5 Georgescu Raluca      raluca.georgescu@email.com   parola5   30/05/24


ID_CATEGORIE NUME_CATEGORIE
------------ --------------------------------------------------
           1 Electronice
           2 Haine
           3 Carti
           4 Jucarii
           5 Electrocasnice


ID_PRODUS NUME                                 PRET ID_CATEGORIE DESCRIERE
--------- ------------------------------ ---------- ------------ ------------------------------------
        1 Laptop Lenovo                        3500            1 Laptop performant pentru birou
        2 Rochie vara                         120.5            2 Rochie din bumbac, marimea M
        3 Harry Potter                           60            3 Carte fantasy populara
        4 Masinuta                               35            4 Jucarie pentru copii peste 3 ani
        5 Mixer Philips                         240            5 Mixer de bucatarie cu 5 viteze
        6 Tabletă Samsung Galaxy Tab A8     1299.99            1 Tabletă 10.5", 4GB RAM, 64GB storage
        7 Cască wireless Sony WH-1000XM4    1299.99            1 Căști cu noise cancelling premium
        8 Smartphone Samsung Galaxy S21     3499.99            1 Telefon flagship cu camera profesională
        9 Tabletă Lenovo Tab P11 Pro        1999.99            1 Tabletă premium cu display OLED

9 rows selected.


ID_COMANDA DATA      ID_CLIENT ADRESA_LIVRARE                         STATUS
---------- -------- ---------- -------------------------------------- --------------------
         1 01/06/24          1 Str. Lalelelor 5, Bucuresti            livrata
         2 03/06/24          2 Bd. Libertatii 10, Timisoara           procesare
         3 05/06/24          3 Str. Florilor 8, Cluj                  anulata
         4 07/06/24          4 Aleea Teiului 18, Iasi                 livrata
         5 09/06/24          5 Str. Zorilor 2, Constanta              procesare
         6 09/07/25          1 Str. Exemplu nr. 1, București          procesare

6 rows selected.


 ID_PRODUS CANTITATE_DISPONIBILA
---------- ---------------------
         1                    10
         2                    22
         3                    15
         4                    30
```

```
 ID_PRODUS CANTITATE_DISPONIBILA
---------- ---------------------
         1                    10
         2                    22
         3                    15
         4                    30
         5                     5
         6                    25
         7                    15
         8                     8
         9                    10

9 rows selected.


  ID_PLATA ID_COMANDA    VALOARE DATA_PLA TIP_PLATA
---------- ---------- ---------- -------- --------------------
         1          1       3500 01/06/24 card
         2          2      120.5 03/06/24 ramburs
         3          3         60 05/06/24 card
         4          4         35 07/06/24 card
         5          5        240 09/06/24 ramburs


ID_COMANDA  ID_PRODUS  CANTITATE PRET_UNITAR
---------- ---------- ---------- -----------
         1          1          1        3500
         2          2          2       120.5
         2          3          1          60
         3          3          1          60
         3          4          2          35
         4          4          2          35
         4          5          1         240
         5          1          1        3500
         5          2          1       120.5
         5          5          1         240
         6          6          2     1299.99

ID_COMANDA  ID_PRODUS  CANTITATE PRET_UNITAR
---------- ---------- ---------- -----------
         1          7          1     1299.99
         2          7          1     1299.99
         1          8          3     3499.99
         2          8          2     3499.99
         3          8          1     3499.99

16 rows selected.
```

```
CREATE SEQUENCE seq_id START WITH 1 INCREMENT BY 1 NOCACHE NOCYCLE;

CREATE TABLE CLIENT (
   id_client NUMBER PRIMARY KEY,
   nume VARCHAR2(50),
   email VARCHAR2(50),=
   parola VARCHAR2(50),
   data_inregistrare DATE
);

CREATE TABLE CATEGORIE (
   id_categorie NUMBER PRIMARY KEY,
   nume_categorie VARCHAR2(50)
);
```

```
CREATE TABLE PRODUS (
   id_produs NUMBER PRIMARY KEY,
   nume VARCHAR2(50),
   pret NUMBER(10,2),
   id_categorie NUMBER,
   descriere VARCHAR2(100),
   CONSTRAINT fk_produs_categorie FOREIGN KEY (id_categorie) REFERENCES
CATEGORIE(id_categorie)
);

CREATE TABLE COMANDA (
   id_comanda NUMBER PRIMARY KEY,
   data DATE,
   id_client NUMBER,
   adresa_livrare VARCHAR2(100),
   status VARCHAR2(20),
   CONSTRAINT fk_comanda_client FOREIGN KEY (id_client) REFERENCES
CLIENT(id_client)
);

CREATE TABLE STOC (
   id_produs NUMBER PRIMARY KEY,
   cantitate_disponibila NUMBER,
   CONSTRAINT fk_stoc_produs FOREIGN KEY (id_produs) REFERENCES
PRODUS(id_produs)
);

CREATE TABLE PLATA (
   id_plata NUMBER PRIMARY KEY,
   id_comanda NUMBER,
   valoare NUMBER(10,2),
   data_plata DATE,
   tip_plata VARCHAR2(20),
   CONSTRAINT fk_plata_comanda FOREIGN KEY (id_comanda) REFERENCES
COMANDA(id_comanda)
);

CREATE TABLE DETALII_COMANDA (
   id_comanda NUMBER,
   id_produs NUMBER,
   cantitate NUMBER,
   pret_unitar NUMBER(10,2),
   PRIMARY KEY (id_comanda, id_produs),
   CONSTRAINT fk_detalii_comanda FOREIGN KEY (id_comanda) REFERENCES
COMANDA(id_comanda),
   CONSTRAINT fk_detalii_produs FOREIGN KEY (id_produs) REFERENCES
PRODUS(id_produs)
```

```
);
-- CATEGORIE
INSERT INTO CATEGORIE VALUES (1, 'Electronice');
INSERT INTO CATEGORIE VALUES (2, 'Haine');
INSERT INTO CATEGORIE VALUES (3, 'Carti');
INSERT INTO CATEGORIE VALUES (4, 'Jucarii');
INSERT INTO CATEGORIE VALUES (5, 'Electrocasnice');

-- CLIENT
INSERT INTO CLIENT VALUES (1, 'Popescu Ana', 'ana.popescu@email.com', 'parola1',
TO_DATE('2024-01-10', 'YYYY-MM-DD'));
INSERT INTO CLIENT VALUES (2, 'Ionescu Mihai', 'mihai.ionescu@email.com', 'parola2',
TO_DATE('2024-02-15', 'YYYY-MM-DD'));
INSERT INTO CLIENT VALUES (3, 'Stan Maria', 'maria.stan@email.com', 'parola3',
TO_DATE('2024-03-20', 'YYYY-MM-DD'));
INSERT INTO CLIENT VALUES (4, 'Dumitru Vlad', 'vlad.dumitru@email.com', 'parola4',
TO_DATE('2024-04-25', 'YYYY-MM-DD'));
INSERT INTO CLIENT VALUES (5, 'Georgescu Raluca', 'raluca.georgescu@email.com',
'parola5', TO_DATE('2024-05-30', 'YYYY-MM-DD'));

-- PRODUS
INSERT INTO PRODUS VALUES (1, 'Laptop Lenovo', 3500.00, 1, 'Laptop performant pentru
birou');
INSERT INTO PRODUS VALUES (2, 'Rochie vara', 120.50, 2, 'Rochie din bumbac, marimea
M');
INSERT INTO PRODUS VALUES (3, 'Harry Potter', 60.00, 3, 'Carte fantasy populara');
INSERT INTO PRODUS VALUES (4, 'Masinuta', 35.00, 4, 'Jucarie pentru copii peste 3 ani');
INSERT INTO PRODUS VALUES (5, 'Mixer Philips', 240.00, 5, 'Mixer de bucatarie cu 5
viteze');

-- COMANDA
INSERT INTO COMANDA VALUES (1, TO_DATE('2024-06-01', 'YYYY-MM-DD'), 1, 'Str.
Lalelelor 5, Bucuresti', 'livrata');
INSERT INTO COMANDA VALUES (2, TO_DATE('2024-06-03', 'YYYY-MM-DD'), 2, 'Bd.
Libertatii 10, Timisoara', 'procesare');
INSERT INTO COMANDA VALUES (3, TO_DATE('2024-06-05', 'YYYY-MM-DD'), 3, 'Str.
Florilor 8, Cluj', 'anulata');
INSERT INTO COMANDA VALUES (4, TO_DATE('2024-06-07', 'YYYY-MM-DD'), 4, 'Aleea
Teiului 18, Iasi', 'livrata');
INSERT INTO COMANDA VALUES (5, TO_DATE('2024-06-09', 'YYYY-MM-DD'), 5, 'Str.
Zorilor 2, Constanta', 'procesare');

-- STOC
INSERT INTO STOC VALUES (1, 10);
INSERT INTO STOC VALUES (2, 22);
INSERT INTO STOC VALUES (3, 15);
INSERT INTO STOC VALUES (4, 30);
INSERT INTO STOC VALUES (5, 5);
```

```
-- PLATA
INSERT INTO PLATA VALUES (1, 1, 3500.00, TO_DATE('2024-06-01', 'YYYY-MM-DD'),
'card');
INSERT INTO PLATA VALUES (2, 2, 120.50, TO_DATE('2024-06-03', 'YYYY-MM-DD'),
'ramburs');
INSERT INTO PLATA VALUES (3, 3, 60.00, TO_DATE('2024-06-05', 'YYYY-MM-DD'),
'card');
INSERT INTO PLATA VALUES (4, 4, 35.00, TO_DATE('2024-06-07', 'YYYY-MM-DD'),
'card');
INSERT INTO PLATA VALUES (5, 5, 240.00, TO_DATE('2024-06-09', 'YYYY-MM-DD'),
'ramburs');

-- DETALII_COMANDA
INSERT INTO DETALII_COMANDA VALUES (1, 1, 1, 3500.00);
INSERT INTO DETALII_COMANDA VALUES (2, 2, 2, 120.50);
INSERT INTO DETALII_COMANDA VALUES (2, 3, 1, 60.00);
INSERT INTO DETALII_COMANDA VALUES (3, 3, 1, 60.00);
INSERT INTO DETALII_COMANDA VALUES (3, 4, 2, 35.00);
INSERT INTO DETALII_COMANDA VALUES (4, 4, 2, 35.00);
INSERT INTO DETALII_COMANDA VALUES (4, 5, 1, 240.00);
INSERT INTO DETALII_COMANDA VALUES (5, 1, 1, 3500.00);
INSERT INTO DETALII_COMANDA VALUES (5, 2, 1, 120.50);
INSERT INTO DETALII_COMANDA VALUES (5, 5, 1, 240.00);

;
```

12.Formulați în limbaj natural și implementați 5 cereri SQL complexe ce vor utiliza, în ansamblul lor, următoarele elemente:

a) subcereri sincronizate în care intervin cel puțin 3 tabele

b) subcereri nesincronizate în clauza FROM

c) grupări de date, funcții grup, filtrare la nivel de grupuri cu subcereri nesincronizate (în clauza de HAVING)

d) ordonări și utilizarea funcțiilor NVL și DECODE (în cadrul aceleiași cereri)

e) utilizarea a cel puțin 2 funcții pe șiruri de caractere, 2 funcții pe date calendaristice, a cel puțin unei expresii CASE

f) utilizarea a cel puțin 1 bloc de cerere (clauza WITH)

```sql
--Cerință: Identificați clienții care au cheltuit peste media totală a tuturor clienților, afișând detaliile lor și suma totală cheltuită, ordonați descrescător după valoarea totală.
-- Elemente utilizate:
-- a) Subcerere sincronizată cu 3 tabele (CLIENT, COMANDA, PLATA)
-- d) Ordonare și funcția NVL
-- c) Grupări de date și filtrare la nivel de grupuri

WITH CheltuieliClienti AS (
    SELECT
        c.id_client,
        c.nume,
        NVL(SUM(p.valoare), 0) AS total_cheltuit
    FROM
        CLIENT c
    LEFT JOIN
        COMANDA co ON c.id_client = co.id_client
    LEFT JOIN
        PLATA p ON co.id_comanda = p.id_comanda
    GROUP BY
        c.id_client, c.nume
)
SELECT
    id_client,
    nume,
    total_cheltuit
FROM
    CheltuieliClienti
WHERE
    total_cheltuit > (SELECT AVG(total_cheltuit) FROM CheltuieliClienti)
ORDER BY
    total_cheltuit DESC;
```

ID_CLIENT	NUME	TOTAL_CHELTUIT
1	Popescu Ana	3500

```sql
--Cerință: Afișați vânzările pe categorii de produse, pe luni, cu detaliere a numărului de produse vândute și valoarea totală, doar pentru categoriile cu vânzări peste media pe categorii.
-- Elemente utilizate:
-- b) Subcerere nesincronizată în FROM
-- c) Grupări și filtrare la nivel de grupuri cu HAVING
-- e) Funcții pe date (EXTRACT, TO_CHAR) și CASE
-- f) Bloc WITH

WITH VanzariCategorii AS (
    SELECT
        cat.nume_categorie,
        TO_CHAR(co.data, 'YYYY-MM') AS luna,
        SUM(dc.cantitate) AS nr_produse,
        SUM(dc.cantitate * dc.pret_unitar) AS valoare_totala
    FROM
        CATEGORIE cat
    JOIN
        PRODUS p ON cat.id_categorie = p.id_categorie
    JOIN
        DETALII_COMANDA dc ON p.id_produs = dc.id_produs
    JOIN
        COMANDA co ON dc.id_comanda = co.id_comanda
    GROUP BY
        cat.nume_categorie, TO_CHAR(co.data, 'YYYY-MM')
)
SELECT
    nume_categorie,
    luna,
    nr_produse,
    valoare_totala,
    CASE
        WHEN valoare_totala > 1000 THEN 'Ridicat'
        WHEN valoare_totala > 500 THEN 'Mediu'
        ELSE 'Scăzut'
    END AS nivel_vanzi
FROM
    VanzariCategorii
WHERE
    nume_categorie IN (
        SELECT nume_categorie
        FROM VanzariCategorii
        GROUP BY nume_categorie
        HAVING SUM(valoare_totala) > (SELECT AVG(valoare_totala) FROM VanzariCategorii)
    )
ORDER BY
    nume_categorie, luna;
```

NUME_CATEGORIE	LUNA	NR_PRODUSE	VALOARE_TOTALA	NIVEL_V
Electronice	2024-06	10	30599.92	Ridicat
Electronice	2025-07	2	2599.98	Ridicat

```sql
--Cerința: Identificați produsele care se află în top 3 în fiecare categorie după valoarea totală a vânzărilor, cu detalierea a stocului rămas.
-- Elemente utilizate:
-- a) Subcerere sincronizată cu 3 tabele (PRODUS, DETALII_COMANDA, CATEGORIE)
-- d) DECODE pentru afișare status stoc
-- e) Funcții pe șiruri (UPPER, SUBSTR)

SELECT
    p.nume AS produs,
    UPPER(SUBSTR(c.nume_categorie, 1, 3)) AS cod_categorie,
    SUM(dc.cantitate * dc.pret_unitar) AS valoare_totala,
    s.cantitate_disponibila,
    DECODE(
        SIGN(s.cantitate_disponibila - 10),
        1, 'Suficient',
        0, 'Limita',
        -1, 'Necesită reaprovizionare'
    ) AS status_stoc
FROM
    PRODUS p
JOIN
    CATEGORIE c ON p.id_categorie = c.id_categorie
JOIN
    DETALII_COMANDA dc ON p.id_produs = dc.id_produs
JOIN
    STOC s ON p.id_produs = s.id_produs
WHERE
    p.id_produs IN (
        SELECT id_produs FROM (
            SELECT
                p.id_produs,
                RANK() OVER (PARTITION BY p.id_categorie ORDER BY SUM(dc.cantitate * dc.pret_unitar) DESC) AS rang
            FROM
                PRODUS p
            JOIN
                DETALII_COMANDA dc ON p.id_produs = dc.id_produs
            GROUP BY
                p.id_produs, p.id_categorie
        )
        WHERE rang <= 3
    )
GROUP BY
    p.nume, c.nume_categorie, s.cantitate_disponibila
ORDER BY
    c.nume_categorie, valoare_totala DESC;
```

PRODUS	COD	VALOARE_TOTALA	CANTITATE_DISPONIBILA	STATUS_STOC
Harry Potter	CAR	120	15	Suficient
Mixer Philips	ELE	480	5	Necesită reaprovizionare
Smartphone Samsung Galaxy S21	ELE	20999.94	8	Necesită reaprovizionare
Laptop Lenovo	ELE	7000	10	Limita
Tabletă Samsung Galaxy Tab A8	ELE	2599.98	25	Suficient
Cască wireless Sony WH-1000XM4	ELE	2599.98	15	Suficient
Rochie vara	HAI	361.5	22	Suficient
Masinuta	JUC	140	30	Suficient

8 rows selected.

2(4).sql > ...

```sql
-- Cerință: Lista clienților cu numărul de comenzi și suma cheltuită, ordonată descrescător
-- Elemente utilizate conform cerinței:
-- a) Subcerere sincronizată implicită (JOIN între CLIENT, COMANDA, PLATA - 3 tabele)
-- c) Grupări de date (GROUP BY) și funcții grup (COUNT, SUM)
-- d) Ordonare (ORDER BY) și funcția NVL
-- e) Funcție pe șiruri (concatenație implicită în SELECT)

SELECT
    c.id_client,
    c.nume AS nume_client,
    c.email,
    COUNT(co.id_comanda) AS numar_comenzi,
    NVL(SUM(p.valoare), 0) AS suma_totala_cheltuita
FROM
    CLIENT c
LEFT JOIN
    COMANDA co ON c.id_client = co.id_client
LEFT JOIN
    PLATA p ON co.id_comanda = p.id_comanda
GROUP BY
    c.id_client, c.nume, c.email
ORDER BY
    suma_totala_cheltuita DESC;
```

ID_CLIENT	NUME_CLIENT	EMAIL	NUMAR_COMENZI	SUMA_TOTALA_CHELTUITA
1	Popescu Ana	ana.popescu@email.com	2	3500
5	Georgescu Raluca	raluca.georgescu@email.com	1	240
2	Ionescu Mihai	mihai.ionescu@email.com	1	120.5
3	Stan Maria	maria.stan@email.com	1	60
4	Dumitru Vlad	vlad.dumitru@email.com	1	35

```sql
-- Cerință: Lista plăților cu detalii despre tipul de plată și statusul comenzii
-- Elemente utilizate:
-- d) Funcțiile NVL și DECODE în cadrul aceleiași cereri
-- e) Funcții pe șiruri de caractere (TO_CHAR pentru formatare)
-- e) Funcții pe date (TO_DATE pentru conversie)

SELECT
    p.id_plata,
    c.nume AS client,
    TO_CHAR(p.data_plata, 'DD-MM-YYYY HH24:MI') AS data_ora_plata,
    p.valoare,

    DECODE(p.tip_plata,
        'card', 'Card bancar',
        'ramburs', 'Ramburs la livrare',
        'transfer', 'Transfer bancar',
        'Alta metoda') AS metoda_plata_descriere,
    NVL(co.status, 'Necunoscut') AS status_comanda,
    DECODE(NVL(co.status, 'Necunoscut'),
        'livrata', 'Comanda finalizata',
        'procesare', 'In curs de procesare',
        'anulata', 'Comanda anulata',
        'Necunoscut') AS status_comanda_descriere
FROM
    PLATA p
JOIN
    COMANDA co ON p.id_comanda = co.id_comanda
JOIN
    CLIENT c ON co.id_client = c.id_client
ORDER BY
    p.data_plata DESC;
```

```
ID_PLATA CLIENT                                            DATA_ORA_PLATA      VALOARE METODA_PLATA_DESCR STATUS_COMANDA       STATUS_COMANDA_DESCR
---------- ------------------------------------------------- ---------------- ---------- ------------------ -------------------- --------------------
         5 Georgescu Raluca                                  09-06-2024 00:00        240 Ramburs la livrare procesare            In curs de procesare
         4 Dumitru Vlad                                      07-06-2024 00:00         35 Card bancar        livrata              Comanda finalizata
         3 Stan Maria                                        05-06-2024 00:00         60 Card bancar        anulata              Comanda anulata
         2 Ionescu Mihai                                     03-06-2024 00:00      120.5 Ramburs la livrare procesare            In curs de procesare
         1 Popescu Ana                                       01-06-2024 00:00       3500 Card bancar        livrata              Comanda finalizata
```

13. Implementarea a 3 operații de actualizare și de suprimare a datelor utilizând subcereri.

```sql
3(1).SQL > ...
  -- Actualizează stocul pentru produsele comandate în luna curentă
  -- Scade cantitatea comandată din stocul disponibil
  UPDATE STOC s
  SET cantitate_disponibila = cantitate_disponibila - (
      SELECT SUM(dc.cantitate)
      FROM DETALII_COMANDA dc
      JOIN COMANDA co ON dc.id_comanda = co.id_comanda
      WHERE dc.id_produs = s.id_produs
      AND EXTRACT(MONTH FROM co.data) = EXTRACT(MONTH FROM SYSDATE)
      AND EXTRACT(YEAR FROM co.data) = EXTRACT(YEAR FROM SYSDATE)
  )
  WHERE EXISTS (
      SELECT 1
      FROM DETALII_COMANDA dc
      JOIN COMANDA co ON dc.id_comanda = co.id_comanda
      WHERE dc.id_produs = s.id_produs
      AND EXTRACT(MONTH FROM co.data) = EXTRACT(MONTH FROM SYSDATE)
      AND EXTRACT(YEAR FROM co.data) = EXTRACT(YEAR FROM SYSDATE)
  );
```

```
1 row updated.
```

```sql
-- Șterge clienții care nu au plasat nicio comandă în ultimele 12 luni
DELETE FROM PLATA
WHERE id_comanda IN (
    SELECT co.id_comanda
    FROM COMANDA co
    WHERE co.id_client IN (
        SELECT c.id_client
        FROM CLIENT c
        WHERE c.id_client NOT IN (
            SELECT DISTINCT co2.id_client
            FROM COMANDA co2
            WHERE co2.data >= ADD_MONTHS(SYSDATE, -12)
        )
        AND c.data_inregistrare < ADD_MONTHS(SYSDATE, -12)
    )
);

DELETE FROM DETALII_COMANDA
WHERE id_comanda IN (
    SELECT co.id_comanda
    FROM COMANDA co
    WHERE co.id_client IN (
        SELECT c.id_client
        FROM CLIENT c
        WHERE c.id_client NOT IN (
            SELECT DISTINCT co2.id_client
            FROM COMANDA co2
            WHERE co2.data >= ADD_MONTHS(SYSDATE, -12)
        )
        AND c.data_inregistrare < ADD_MONTHS(SYSDATE, -12)
    )
);

DELETE FROM COMANDA
WHERE id_client IN (
    SELECT c.id_client
    FROM CLIENT c
    WHERE c.id_client NOT IN (
        SELECT DISTINCT co.id_client
        FROM COMANDA co
        WHERE co.data >= ADD_MONTHS(SYSDATE, -12)
    )
    AND c.data_inregistrare < ADD_MONTHS(SYSDATE, -12)
);

DELETE FROM CLIENT
WHERE id_client NOT IN (
    SELECT DISTINCT co.id_client
    FROM COMANDA co
    WHERE co.data >= ADD_MONTHS(SYSDATE, -12)
)
AND data_inregistrare < ADD_MONTHS(SYSDATE, -12);
```

```
4 rows deleted.


12 rows deleted.


4 rows deleted.


4 rows deleted.
```

```sql
-- Actualizare promoțională a prețurilor pentru produsele din categoria 'Electronice'
-- care au vânzări sub media categoriei (reducere 15%)
UPDATE PRODUS
SET pret = pret * 0.85
WHERE id_categorie = (
    SELECT id_categorie
    FROM CATEGORIE
    WHERE nume_categorie = 'Electronice'
)
AND id_produs IN (
    SELECT p.id_produs
    FROM PRODUS p
    LEFT JOIN DETALII_COMANDA dc ON p.id_produs = dc.id_produs
    WHERE p.id_categorie = (
        SELECT id_categorie
        FROM CATEGORIE
        WHERE nume_categorie = 'Electronice'
    )
    GROUP BY p.id_produs
    HAVING NVL(SUM(dc.cantitate), 0) < (
        SELECT AVG(sums.suma_cantitati)
        FROM (
            SELECT SUM(dc2.cantitate) as suma_cantitati
            FROM PRODUS p2
            LEFT JOIN DETALII_COMANDA dc2 ON p2.id_produs = dc2.id_produs
            WHERE p2.id_categorie = (
                SELECT id_categorie
                FROM CATEGORIE
                WHERE nume_categorie = 'Electronice'
            )
            GROUP BY p2.id_produs
        ) sums
    )
);
```

```
4 rows updated.


4 rows updated.
```

--Cerință: Identificați clienții care au cheltuit peste media totală a tuturor clienților, afișând detaliile lor și suma totală cheltuită, ordonați descrescător după valoarea totală.

-- Elemente utilizate:

-- a) Subcerere sincronizată cu 3 tabele (CLIENT, COMANDA, PLATA)

-- d) Ordonare și funcția NVL

-- c) Grupări de date și filtrare la nivel de grupuri

```sql
WITH CheltuieliClienti AS (
    SELECT
        c.id_client,
        c.nume,
        NVL(SUM(p.valoare), 0) AS total_cheltuit
    FROM
        CLIENT c
    LEFT JOIN
```

```
    COMANDA co ON c.id_client = co.id_client
  LEFT JOIN
    PLATA p ON co.id_comanda = p.id_comanda
  GROUP BY
    c.id_client, c.nume
)
SELECT
  id_client,
  nume,
  total_cheltuit
FROM
  CheltuieliClienti
WHERE
  total_cheltuit > (SELECT AVG(total_cheltuit) FROM CheltuieliClienti)
ORDER BY
  total_cheltuit DESC;
--Cerință: Afișați vânzările pe categorii de produse, pe luni, cu detaliere a numărului
de produse vândute și valoarea totală, doar pentru categoriile cu vânzări peste media
pe categorii.
-- Elemente utilizate:
-- b) Subcerere nesincronizată în FROM
-- c) Grupări și filtrare la nivel de grupuri cu HAVING
-- e) Funcții pe date (EXTRACT, TO_CHAR) și CASE
-- f) Bloc WITH

WITH VanzariCategorii AS (
  SELECT
    cat.nume_categorie,
    TO_CHAR(co.data, 'YYYY-MM') AS luna,
    SUM(dc.cantitate) AS nr_produse,
    SUM(dc.cantitate * dc.pret_unitar) AS valoare_totala
  FROM
    CATEGORIE cat
  JOIN
    PRODUS p ON cat.id_categorie = p.id_categorie
  JOIN
    DETALII_COMANDA dc ON p.id_produs = dc.id_produs
  JOIN
    COMANDA co ON dc.id_comanda = co.id_comanda
  GROUP BY
    cat.nume_categorie, TO_CHAR(co.data, 'YYYY-MM')
)
SELECT
  nume_categorie,
  luna,
  nr_produse,
  valoare_totala,
  CASE
```

```
      WHEN valoare_totala > 1000 THEN 'Ridicat'
      WHEN valoare_totala > 500 THEN 'Mediu'
      ELSE 'Scăzut'
   END AS nivel_vanzi
FROM
   VanzariCategorii
WHERE
   nume_categorie IN (
      SELECT nume_categorie
      FROM VanzariCategorii
      GROUP BY nume_categorie
      HAVING SUM(valoare_totala) > (SELECT AVG(valoare_totala) FROM
VanzariCategorii)
   )
ORDER BY
   nume_categorie, luna;
--Cerință: Identificați produsele care se află în top 3 în fiecare categorie după valoarea
totală a vânzărilor, cu detaliere a stocului rămas.
-- Elemente utilizate:
-- a) Subcerere sincronizată cu 3 tabele (PRODUS, DETALII_COMANDA, CATEGORIE)
-- d) DECODE pentru afișare status stoc
-- e) Funcții pe șiruri (UPPER, SUBSTR)

SELECT
   p.nume AS produs,
   UPPER(SUBSTR(c.nume_categorie, 1, 3)) AS cod_categorie,
   SUM(dc.cantitate * dc.pret_unitar) AS valoare_totala,
   s.cantitate_disponibila,
   DECODE(
      SIGN(s.cantitate_disponibila - 10),
      1, 'Suficient',
      0, 'Limita',
      -1, 'Necesită reaprovizionare'
   ) AS status_stoc
FROM
   PRODUS p
JOIN
   CATEGORIE c ON p.id_categorie = c.id_categorie
JOIN
   DETALII_COMANDA dc ON p.id_produs = dc.id_produs
JOIN
   STOC s ON p.id_produs = s.id_produs
WHERE
   p.id_produs IN (
      SELECT id_produs FROM (
         SELECT
            p.id_produs,
```

```
            RANK() OVER (PARTITION BY p.id_categorie ORDER BY SUM(dc.cantitate *
dc.pret_unitar) DESC) AS rang
         FROM
            PRODUS p
         JOIN
            DETALII_COMANDA dc ON p.id_produs = dc.id_produs
         GROUP BY
            p.id_produs, p.id_categorie
      )
      WHERE rang <= 3
   )
GROUP BY
   p.nume, c.nume_categorie, s.cantitate_disponibila
ORDER BY
   c.nume_categorie, valoare_totala DESC;
-- Cerință: Lista clienților cu numărul de comenzi și suma cheltuită, ordonată
descrescător
-- Elemente utilizate conform cerinței:
-- a) Subcerere sincronizată implicita (JOIN între CLIENT, COMANDA, PLATA - 3
tabele)
-- c) Grupări de date (GROUP BY) și funcții grup (COUNT, SUM)
-- d) Ordonare (ORDER BY) și funcția NVL
-- e) Funcție pe șiruri (concatenație implicită în SELECT)

SELECT
   c.id_client,
   c.nume AS nume_client,
   c.email,
   COUNT(co.id_comanda) AS numar_comenzi,
   NVL(SUM(p.valoare), 0) AS suma_totala_cheltuita
FROM
   CLIENT c
LEFT JOIN
   COMANDA co ON c.id_client = co.id_client
LEFT JOIN
   PLATA p ON co.id_comanda = p.id_comanda
GROUP BY
   c.id_client, c.nume, c.email
ORDER BY
   suma_totala_cheltuita DESC;
-- Cerință: Lista plăților cu detalii despre tipul de plată și statusul comenzii
-- Elemente utilizate:
-- d) Funcțiile NVL și DECODE în cadrul aceleiași cereri
-- e) Funcții pe șiruri de caractere (TO_CHAR pentru formatare)
-- e) Funcții pe date (TO_DATE pentru conversie)

SELECT
   p.id_plata,
```

```
    c.nume AS client,
    TO_CHAR(p.data_plata, 'DD-MM-YYYY HH24:MI') AS data_ora_plata,
    p.valoare,

    DECODE(p.tip_plata,
        'card', 'Card bancar',
        'ramburs', 'Ramburs la livrare',
        'transfer', 'Transfer bancar',
        'Alta metoda') AS metoda_plata_descriere,
    NVL(co.status, 'Necunoscut') AS status_comanda,
    DECODE(NVL(co.status, 'Necunoscut'),
        'livrata', 'Comanda finalizata',
        'procesare', 'In curs de procesare',
        'anulata', 'Comanda anulata',
        'Necunoscut') AS status_comanda_descriere
FROM
    PLATA p
JOIN
    COMANDA co ON p.id_comanda = co.id_comanda
JOIN
    CLIENT c ON co.id_client = c.id_client
ORDER BY
    p.data_plata DESC;
-- Actualizează stocul pentru produsele comandate în luna curentă
-- Scade cantitatea comandată din stocul disponibil
UPDATE STOC s
SET cantitate_disponibila = cantitate_disponibila - (
    SELECT SUM(dc.cantitate)
    FROM DETALII_COMANDA dc
    JOIN COMANDA co ON dc.id_comanda = co.id_comanda
    WHERE dc.id_produs = s.id_produs
    AND EXTRACT(MONTH FROM co.data) = EXTRACT(MONTH FROM SYSDATE)
    AND EXTRACT(YEAR FROM co.data) = EXTRACT(YEAR FROM SYSDATE)
)
WHERE EXISTS (
    SELECT 1
    FROM DETALII_COMANDA dc
    JOIN COMANDA co ON dc.id_comanda = co.id_comanda
    WHERE dc.id_produs = s.id_produs
    AND EXTRACT(MONTH FROM co.data) = EXTRACT(MONTH FROM SYSDATE)
    AND EXTRACT(YEAR FROM co.data) = EXTRACT(YEAR FROM SYSDATE)
);
-- Șterge clienții care nu au plasat nicio comandă în ultimele 12 luni
DELETE FROM PLATA
WHERE id_comanda IN (
    SELECT co.id_comanda
    FROM COMANDA co
    WHERE co.id_client IN (
```

```
      SELECT c.id_client
      FROM CLIENT c
      WHERE c.id_client NOT IN (
        SELECT DISTINCT co2.id_client
        FROM COMANDA co2
        WHERE co2.data >= ADD_MONTHS(SYSDATE, -12)
      )
      AND c.data_inregistrare < ADD_MONTHS(SYSDATE, -12)
    )
);

DELETE FROM DETALII_COMANDA
WHERE id_comanda IN (
    SELECT co.id_comanda
    FROM COMANDA co
    WHERE co.id_client IN (
      SELECT c.id_client
      FROM CLIENT c
      WHERE c.id_client NOT IN (
        SELECT DISTINCT co2.id_client
        FROM COMANDA co2
        WHERE co2.data >= ADD_MONTHS(SYSDATE, -12)
      )
      AND c.data_inregistrare < ADD_MONTHS(SYSDATE, -12)
    )
);

DELETE FROM COMANDA
WHERE id_client IN (
    SELECT c.id_client
    FROM CLIENT c
    WHERE c.id_client NOT IN (
      SELECT DISTINCT co.id_client
      FROM COMANDA co
      WHERE co.data >= ADD_MONTHS(SYSDATE, -12)
    )
    AND c.data_inregistrare < ADD_MONTHS(SYSDATE, -12)
);

DELETE FROM CLIENT
WHERE id_client NOT IN (
    SELECT DISTINCT co.id_client
    FROM COMANDA co
    WHERE co.data >= ADD_MONTHS(SYSDATE, -12)
)
AND data_inregistrare < ADD_MONTHS(SYSDATE, -12);
-- Actualizare promoțională a prețurilor pentru produsele din categoria 'Electronice'
-- care au vânzări sub media categoriei (reducere 15%)
```

```
UPDATE PRODUS
SET pret = pret * 0.85
WHERE id_categorie = (
   SELECT id_categorie
   FROM CATEGORIE
   WHERE nume_categorie = 'Electronice'
)
AND id_produs IN (
   SELECT p.id_produs
   FROM PRODUS p
   LEFT JOIN DETALII_COMANDA dc ON p.id_produs = dc.id_produs
   WHERE p.id_categorie = (
      SELECT id_categorie
      FROM CATEGORIE
      WHERE nume_categorie = 'Electronice'
   )
   GROUP BY p.id_produs
   HAVING NVL(SUM(dc.cantitate), 0) < (
      SELECT AVG(sums.suma_cantitati)
      FROM (
         SELECT SUM(dc2.cantitate) as suma_cantitati
         FROM PRODUS p2
         LEFT JOIN DETALII_COMANDA dc2 ON p2.id_produs = dc2.id_produs
         WHERE p2.id_categorie = (
            SELECT id_categorie
            FROM CATEGORIE
            WHERE nume_categorie = 'Electronice'
         )
         GROUP BY p2.id_produs
      ) sums
   )
);
```